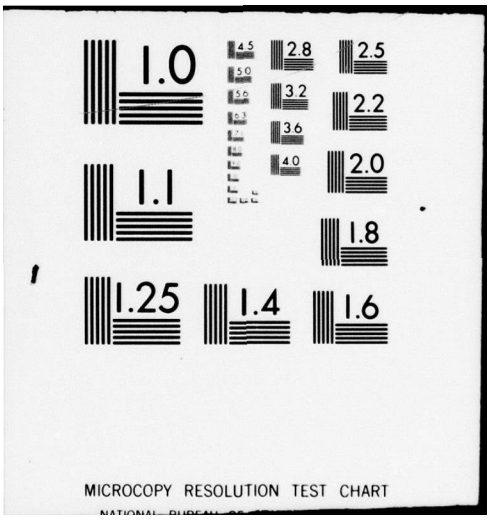
AD-A063 388 SOUTHERN METHODIST UNIV DALLAS TX DEPT OF COMPUTER S--ETC F/G 9/5
SEPARATING AND COMPLETELY SEPARATING SYSTEMS AND LINEAR CODES.(U)
AUG 78 B BOSE, T R RAO N00014-77-C-0455
UNCLASSIFIED CS-7813 NL

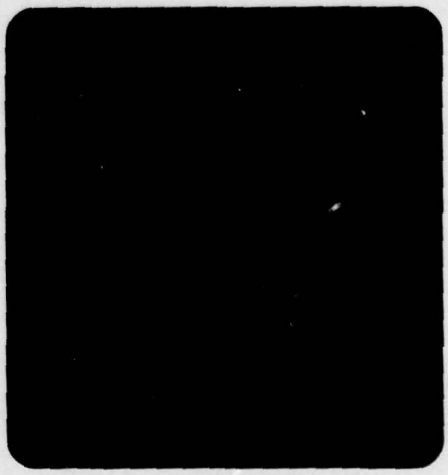
END
DATE
FILMED
3-79
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART

LEVEL II



# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING





SOUTHERN METHODIST UNIVERSITY
SCHOOL OF ENGINEERING
AND APPLIED SCIENCE
DALLAS, TEXAS 75275

LEVEL II

(9) Technical Report CS-7813

# (6) SEPARATING AND COMPLETELY SEPARATING SYSTEMS AND LINEAR CODES*.

(14) Bella Bose
T. R. N. Rao

(12) 17p

(14) CS-7813

DDC
RECEIVED
JAN 18 1979
A

Department of Computer Science
Southern Methodist University
Dallas, Texas

(15) N00014-77-C-0455
NSF-ENG76-11237

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

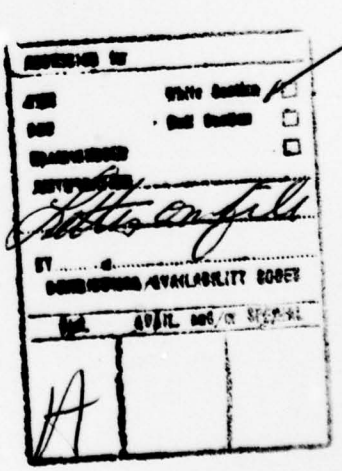
(11) August 1978

*Supported by NSF grant ENG 76-11237 and ONR Contract N00014-77-0455.

409 158

## Abstract

In this report we establish the necessary and sufficient conditions for a set of vectors to be a separating system (SS) or a completely separating system (CSS) from coding theory framework. Then we show that in the case of linear codes the necessary and sufficient condition required for (1,1) CSS is similar to that of (2,1) SS and by deleting the $\underline{0}$ vector from a binary code that forms a (2,1) SS, the set of remaining code words forms a (1,1) CSS. Even though some linear codes form (2,1) and (2,2) SS, we prove here that no linear code forms a (2,1) or a (2,2) CSS.

Index Terms:
Asynchronous circuit, completely separating system, critical race, linear code, ordered pair, separating system, unicode single transition time assignment, unordered pair.

## I. INTRODUCTION

Sequential circuits are commonly classified as being either synchronous or asynchronous. An asynchronous sequential circuit differs from a synchronous sequential circuit in that it contains no clock pulses which regulate the circuit. The advantage of asynchronous circuit is that its circuit response could be faster than that of synchronous circuit. This is because the asynchronous sequential circuit does not have to wait for the arrival of clock pulses before effecting a transition.

Since the circuit terminal action for synchronous case is examined only when a clock pulse appears, the transient conditions during the change of state variables can be completely ignored and several state variables are allowed to change simultaneously. However, for asynchronous case circuit action is examined at all times. If more than one secondary variable is allowed to change then this is called a race. If the final state which the circuit has reached does not depend on the order in which the variables change, then the race is said to be "noncritical race". If the final state reached by the circuit depends on the order in which the internal variables change, then this is referred to as a "critical race".

Critical races must be avoided in asynchronous sequential circuit. This problem can be handled by restricting the state assignments in such a manner that there are no state transitions which involve critical races. A class of state assignments called "unicode single transition time" (STT) assignments were first developed by Liu [2] and later extended by Tracey [6] for avoiding critical races. In these assignments all variables which must change in a given transition are allowed to change simultaneously without critical races. Friedman et al. [1] studied the same problem and showed how (2,2) and (2,1)

separating systems correspond to state assignments for asynchronous circuits.

Sometimes it may be desirable to design a sequential circuit in such a manner that all next state functions to be unate*[8-12]. Mago [4] studied this problem and showed the usefulness of completely separating systems (CSS) ((1,1),(2,1) and (2,2) CSS) for sequential circuit state assignments.

Recently Pradhan and Reddy [5] have given techniques to construct (2,1) SS from linear codes. In this report some more properties of linear codes for state assignments are derived. In Section II we cover some background material. In Section III we establish certain necessary and sufficient conditions for (2,2), (2,1) and (1,1) SS and CSS using coding theory framework. Then we show by omitting the $\underline{0}$ vector from a linear code which forms a (2,1) SS, the set of remaining code words forms a (1,1) SS. Then we show that no linear code forms (2,1) or (2,2) CSS.

## II. DEFINITIONS AND REVIEW OF EARLIER WORK

Friedman et al. [1] generalized the concept of separating systems as follows:

Definition 2.1: Let H be a finite set and $A_1, A_2, \ldots A_n$ be subsets of H. $A = \{A_1 A_2 \ldots A_n\}$ is called an (i,j) separating system of H if for any two subsets R and S of H that have the property $|R|=i$, $|S| = j$ and $R \cap S = \phi$. there exists an $A_k (1 \leq k \leq n)$ such that either

$$\left.\begin{array}{l} R \subseteq A_k \text{ and } S \cap A_k = \phi \\ \text{or} \\ S \subseteq A_k \text{ and } R \cap A_k = \phi. \end{array}\right\} \qquad (2.1)$$

The concept of completely separating systems is generalized by Mago [4] as follows.

---

*A function f is said to be unate iff no variables appear both uncomplemented and complemented when f is written in a minimal sum of products (or product of sums) form.

Definition 2.2: Let H be a finite set and $A_1, A_2, \ldots A_n$ be subsets of H. $A = \{A_1, A_2, \ldots A_n\}$ is called an (i,j) completely separating systems of H if for every two subsets R and S of H, that have the property of $|R|=i$, $|S|=j$ and $R \cap S = \phi$, there exists $A_k$ and $A_\ell$ ($1 \leq k, \ell \leq n$) such that

$$R \subseteq A_k \text{ and } S \cap A_k = \phi$$
and (2.2)
$$S \subseteq A_i \text{ and } R \cap A_i = \phi \ .$$

The relationship between SS and state assignment can be explained as follows. Let the $A = \{A_1, A_2, \ldots A_n\}$ forms an (i,j) SS of H. Then any state $r \in H$ is assigned a binary n-tuple $(y_1, y_2, \ldots y_n)$ where each $y_i = 1$ (or 0) iff $r \in A_i$.

For an example let $H_1 = \{a,b,c,d\}$, $A_1 = \{a,b\}$ and $A_2 = \{a,c\}$. The set $\{A_1, A_2\}$ forms a (1,1) SS of $H_1$. The corresponding state assignment for the elements in H is as follows:

| | $y_1$ | $y_2$ |
|---|---|---|
| a | 1 | 1 |
| b | 1 | 0 |
| c | 0 | 1 |
| d | 0 | 0 |

One can readily observe the association of $A_1$ with $y_1$ and $A_2$ with $y_2$ in the above example. As another example consider the set $H_2 = \{a,b,c\}$. When $A_1 = \{b,c\}$, $A_2 = \{a,c\}$ and $A_3 = \{a,b\}$ then the set $\{A_1, A_2, A_3\}$ forms a (1,1) CSS of $H_2$. The corresponding state assignment is given below

| | $y_1$ | $y_2$ | $y_3$ |
|---|---|---|---|
| a | 0 | 1 | 1 |
| b | 1 | 0 | 1 |
| c | 1 | 1 | 0 |

It can be easily observed that (1,1) SS corresponds to any arbitrary state assignment (i.e. each state assigned to a unique n-tuple). It is shown previously in [1,3] how (2,1) SS and (2,2) SS enable state assignments for asynchronous circuits free from critical races. Further, Mago [4] showed how [1,1] CSS corresponds to a state assignment for a synchronous circuit which results in unate next state functions.

We use the definition given by Pradhan and Reddy [5] for (2,1) SS and extend this definition to other types of SS and CSS. These can be derived from definitions 2.1and 2.2 by the state assignment method explained above. All n-tuples referred to in this report are also called vectors and they are over the binary field {0,1}.

Definition 2.3: Let $X = (x_1 x_2 \cdots x_n)$ and $Y = (y_1 y_2 \cdots y_n)$ be two n-tuples. The transition path from X to Y is the set of all n-tuples obtained by arbitrarily specifying the entries in the positions in which X and Y differ.

Example Let X = 1101 and Y = 1000. Since X and Y differ in second and fourth positions the transition path from X to Y is

$T_{xy}$ = {1101, 1100, 1001, 1000}.

Now (2,1) SS and (2,2) SS can be specified as follows.

Definition 2.4: A set of vectors A of n-tuples is a (2,1) SS iff for all distinct X, Y, Z $\varepsilon$ A, Z does not exist in the transition path from X to Y (i.e. $Z \notin T_{xy}$).

Definition 2.5: A set A of n-tuples is (2,2) SS iff for all distinct U, V, X, Y $\varepsilon$ A, the transition path from U to V and from X to Y are mutually exclusive (i.e. $T_{uv} \cap T_{xy} = \phi$).

Definition 2.6: An n-tuple $X = (x_1 x_2 \cdots x_n)$ is said to cover the n-tuple $Y = (y_1 y_2 \cdots y_n)$ whenever $y_i = 1$ then $x_i = 1$. (X covers Y is written as $Y \leq X$.) Also if $X \leq Y$ or $Y \leq X$ than these vectors are called ordered vectors. If $X \not\leq Y$ and $Y \not\leq X$ then these vectors are called unorderd (or an unordered pair).

Now we can redefine (1,1) CSS, (2,1) CSS and (2,2) CSS as follows.

Definition 2.7: A set A of n-tuples is a (1,1) CSS iff for all $X,Y \in A$, $X \not\leq Y$ and $Y \not\leq X$.

Definition 2.8: A set A of n-tuples is a (2,1) CSS iff for all $X,Y,Z \in A$, $X \neq Z$ and $Y \neq Z$, Z does not exist in the transition path from X to Y (i.e. $Z \notin T_{xy}$) and for all $W \in T_{xy}$, Z and W are unordered vectors.

Definition 2.9: A set of vectors A of n-tuples is a (2,2) CSS iff for all distinct $U,V,X,Y \in A$, the transition paths from U to V and from X to Y are mutually exclusive (i.e. $T_{xy} \cap T_{uv} = \phi$) and all pair R and S are unordered where $R \in T_{uv}$ and $S \in T_{xy}$.

Notation: We adopt the standard notation for logical operators AND, EXOR (+) and NEGATION of n-tuples as follows:

$$X \cdot Y = (x_1 \cdot y_1, x_2 \cdot y_2, \ldots, x_n \cdot y_n)$$

$$X + Y = (x_1 + y_1, x_2 + y_2 \cdots x_n + y_n)$$

and $\bar{X} = (\bar{x}_1, \bar{x}_2 \cdots \bar{x}_n)$

Also let $\underline{0} = (0,0 \cdots 0)$.

A set of n-tuples, C, forming a linear code are a subspace of the vector space of all n-tuples. However, for the binary case one can easily show that C needs only to be closed under addition ( + ) operation. Therefore we state the following.

Definition 2.10: A set of n-tuples A is a linear code iff for all $X,Y \in A$, $X + Y \in A$.

(Note: The operation '+' is modulo 2 addition and for any n-tuple X, $X + X = \underline{0}$.)

## III. LINEAR CODES, SEPARATING SYSTEMS AND COMPLETELY SEPARATING SYSTEMS

In this section the necessary and sufficient conditions for a set of vectors to be (2,1) SS, (2,2) SS, (1,1) CSS, (2,1) CSS and (2,2) CSS are established. Then the relationship between linear codes and SS (CSS) is established. The relationship between (2,1) SS and linear codes has already been given by Pradhan and Reddy in [5] and the following two lemmas are from their work.

Lemma 3.1: A set of vectors A, forms a (2,1) separating system iff for all distinct $X,Y,Z \in A$, $(X + Z) \cdot (Y + Z) \neq 0$.

Lemma 3.2: A linear code C is a (2,1) separating system iff for all non-zero $X,Y \in C$, $X \cdot Y \neq 0$.

The following Lemmas 3.3 and 3.4 give the necessary and sufficient conditions for a set of vectors to be a (2,2) separating system.

Lemma 3.3: A set of vectors A is a (2,2) separating system iff for all distinct $U,V,X,Y \in A$, U and V have the same value e in some position, say i, and X and Y have the complement value $\bar{e}$ in position i.

Proof: Suppose for all distinct $U,V,X,Y \in A$ there is some position j such that $u_j = v_j = e$ and $x_j = y_j = \bar{e}$, then all the elements in the transition path $T_{uv}$ have value e in position i and all elements in the transition path $T_{xy}$ have value $\bar{e}$ in position i. Therefore $T_{uv} \cap T_{xy} = \phi$ and hence A forms a (2,2) SS.

Conversely, let A be a (2,2) SS. If there exists $U,V,X,Y \in A$ such that in no position of U,V,X and Y, $x_i = y_i = e$ and $u_i = v_i = \bar{e}$, then it is easy to prove $T_{uv} \cap T_{xy} \neq \phi$. This contradicts the hypothesis that A is a (2,2) SS. This completes the proof.

Lemma 3.4: A set of vectors A is a (2,2) SS iff for all distinct $U,V,X,Y \in A$ we have $(X + U) \cdot (Y + U) \cdot (X + V) \neq \underline{0}$.

Proof: Suppose for all distinct $u,v,x,y \in A$ if we have $(X + U) \cdot (Y + U)$ $(X + V) \neq \underline{0}$ then at least in one position, say j, we will have

$$x_j + u_j = 1$$
$$y_j + u_j = 1$$
$$x_j + v_j = 1$$

Hence $x_j = y_j = e$ and $u_j = v_j = \bar{e}$ where $e \in \{0,1\}$ is satisfied. Therefore from Lemma 3.4, A is a (2,2) SS.

Conversely if A is a (2,2) SS, then from Lemma 3.3, for all distinct U,V,X,Y in A there exists at least one position, say i, such that

$$x_i = y_i = e \text{ and } u_i = v_i = \bar{e}.$$

Then

$$x_i + u_i = 1,\ y_i + u_i = 1 \text{ and } x_i + v_i = 1.$$

Hence $(X + U) \cdot (Y + U) \cdot (X + V) \neq \underline{0}$.

The following theorem gives the necessary and sufficint conditions for a linear code to be a (2,2) SS.

Theorem 3.5: A linear code C is a (2,2) SS iff all non-zero $P,Q,R \in C$, and $R \neq P + Q$, satisfy

$$P \cdot Q \cdot R \neq \underline{0}.$$

Proof: Let C be a linear code and a (2,2) SS. For all distinct $U,V,X,Y \in C$, a linear code, $U + V = P$, $Y + U = Q$, and $X + U = R$ are all in C. Further P, Q, R are all non-zero since for binary n-tuples, each is its own (unique) inverse. Also $P + Q = U + V + Y + U = V + U$ and $V + U \neq X + U = R$. Therefore $P + Q \neq R$. Finally, since C is a (2,2) SS, Lemm 3.4 holds, which here translates to $P \cdot Q \cdot R \neq 0$.

Conversely, let C be a linear code, satisfying the condition that all non-zero $P, Q, R \in C$, $P + Q \neq R$, and $P, Q, R \neq 0$. Then we need to show that C is

a (2,2) SS. For that purpose, consider any distinct codewords U,V,X,Y $\epsilon$ C. Then X + U = L, Y + U = M, X + U = N are also codewords and L + M = X + U + Y + U = X + Y $\neq$ N. By our hypothesis, L.M.N $\neq$ 0 and therefore by virtue of Lemma 3.4, C must be a (2,2) SS. That completes the proof.

At this point one can easily verify the equidistance codes used by Liu in [2], for state assignment satisfy the conditions stated in Theorem 3.5. This method requires $2^n$-1 secondary state variables for an asynchronous circuit with $2^n$ states. The above theorem could be used to derive linear codes which form (2,2) SS and which may perhaps result in fewer state variables.

Let us now consider the case for completely separating systems. Lemma 3.6 gives necessary and sufficient conditions for a set of vectors to be a (1,1) CSS.

Lemma 3.6: A set of vectors A, forms a (1,1) CSS iff for all distinct X, Y $\epsilon$ A X $\cdot$ (X + Y) $\neq$ 0 and Y $\cdot$ (X + Y) $\neq$ 0.

Proof: For all X,Y $\epsilon$ A let X(X + Y) $\neq$ 0 and Y $\cdot$ (X + Y) $\neq$ 0. Since X $\cdot$ (X + Y) $\neq$ 0 then at some position, say i, $x_i$ = 1 and $y_i$ = 0. Hence X $\not\leq$ Y. Similarly Y $\cdot$(X + Y) $\neq$ 0 leads to Y $\not\leq$ X. Therefore A is a (1,1) CSS.

Conversely, let there be X, Y in C such that X $\cdot$ (X + Y) = $\underline{0}$. Now whenever $x_i$ = 1, then $y_i$ = 1, i.e. X $\leq$ Y, which violates the definition of (1,1) CSS. Similar result can be proved when Y $\cdot$ (X + Y) = 0. That completes the proof.

Theorem 3.7: Let C be a linear code and C' be a set of all non-zero code vectors. C' is a (1,1) CSS iff for all distinct X,Y $\epsilon$ C',X $\cdot$ Y $\neq$ 0.

Proof: From Lemma 3.6, C' is a (1,1) CSS iff for all distinct X, Y $\epsilon$ C', X $\cdot$ (X + Y) $\neq$ 0 and Y $\cdot$ (Y + X) $\neq$ 0. Since C is linear code for X $\neq$ Y, X + Y = Z $\neq$ 0 and hence the theorem.

From Lemma 3.2 and Theorem 3.7 it can be said that in the case of linear codes, the (2,1) SS and (1,1) CSS are in a sense equivalent, i.e. the linear codes which form (2,1) SS also form (1,1) CSS by simply deleting the $\underline{0}$ vector. Hence the codes given by Pradhan and Reddy in [5] could be used to construct (1,1) CSS. This may not give a minimum number of secondary variables. It is known [13] that Berger codes form (1,1) CSS which requires only $n+\log_2(n+1)$ secondary state variables for a flow table with $2^n$ states.

Lemma 3.8: A set of vectors A is a (2,1) completely separating system iff for all distinct X, Y, Z $\epsilon$ A, there exists two positions, say i and j such that $x_i = y_i = e$, $z_i = \bar{e}$ and $x_j = y_j = \bar{e}$, $z_j = e$ where $e \epsilon \{0,1\}$.

Proof: Immediate consequence of Definition 2.8.

Lemma 3.9: A set of vectors A is a (2,1) CSS iff for all distinct X,Y, Z $\epsilon$ A $X \cdot (X + Z) \cdot (Y + Z) \neq 0$, and $Z \cdot (X + Z) \cdot (Y + Z) \neq 0$.

Proof: Suppose for all distinct X, Y, Z $\epsilon$ A, if $X \cdot (X + Z) \cdot (Y + Z) \neq 0$, then at least in one position say i, $x_i = 1$, $y_i = 1$ and $z_i = 0$. Also, for all distinct X, Y, Z $\epsilon$ A, if $Z \cdot (X + Z) \cdot (Y + Z) \neq 0$, then at least in one position say j, $x_j = y_j = 0$ and $z_j = 1$. Therefore from Lemma 3.8, A is a (2,1) CSS.

Conversely, let there be X, Y, Z in A such that $X \cdot (X + Z) \cdot (Y + Z) = 0$. Then whenever $x_i = 1$, $z_i = 1$ and $y_i = \phi$ (don't care) or $z_i = 0$ and $y_i = 0$. From Lemma 3.8 we can see this violates the condition ($x_i = y_i = e$ and $z_i = \bar{e}$) required for A to be a (2,1) CSS. Similar result can be proved when $Z \cdot (X+Z) \cdot (Y+Z) = 0$.

The following lemma follows directly from Definition 2.9.

Lemma 3.10: A set of vectors A is a (2,2) CSS iff for all distinct U, V, X, Y ε A, there exists some positions, say i and j, where

$$u_i = v_i = e,\ x_i = y_i = e$$

and

$$u_j = v_j = \bar{e},\ x_j = y_j = e .$$

Lemma 3.11: A set of vectors A is a (2,2) CSS iff for all distinct U,V,X,Y ε A, $U \cdot (U + X) \cdot (V + X) \cdot (U + Y) \neq \underline{0}$ and $Y \cdot (U + X) \cdot (V + X) \cdot (U + Y) = \underline{0}$.

This lemma can be proved similar to Lemma 3.9.

Theorem 3.12: No linear code forms a (2,1) CSS or a (2,2) CSS.

Proof: From Lemma 3.9, we can see for a linear code C to be a (2,1) CSS, for all distinct X, Y, Z ε C, $X \cdot (X + Z) \cdot (Y + Z) \neq \underline{0}$ and $Z \cdot (X + Z) \cdot (Y + Z) \neq \underline{0}$. But for X, Y, Z ε C, where X and Y are nonzero and distinct, and Z = X + Y, $Z \cdot (X + Z) \cdot (Y + Z) = (X + Y) \cdot (X + X + Y) \cdot (Y + X + Y) = (X + Y) \cdot XY = X \cdot X \cdot Y + Y \cdot X \cdot Y = XY + XY = \underline{0}$. This violates the condition given in Lemma 3.9 for C to be a (2,1) CSS. Therefore C does not form a (2,1) CSS.

It was shown in [4] that a (2,2) CSS is also a (2,1) CSS. If a linear code C forms a (2,2) CSS, then it also forms a (2,1) CSS. This contradicts the frist part of the theorem. Therefore no linear code forms a (2,2) CSS.

## IV. CONCLUSION

In this report we have established certain necessary and sufficient conditions for different types of separating systems and completely separating systems that could be of value in the design of synchronous and asynchronous circuits. We have shown that linear codes that form (2,1) SS can also be used as (1,1) CSS simply by omitting the $\underline{0}$ codeword. While some linear codes can be used to form (2,1) SS and (2,2) SS, we have shown that linear codes cannot be used to form (2,1) CSS or (2,2) CSS. Therefore we direct our future efforts in forming these CSS from non-linear codes, such as coset codes [14] and group theoretical codes [15,16].

References

1. A. D. Friedman, R. L. Graham and J. D. Ullman, "Universal Single Transition Time Asynchronous State Assignments", IEEE Trans. Comput. Vol. C-18, pp. 541-547, June 1969.

2. C. N. Liu, "A State Variable Assignment Method for Asynchronous Sequential Circuits", J. ACM, Vol. 10, pp. 209-216, April 1963.

3. G. Mago, "Realization of Methods for Asynchronous Sequential Circuits", IEEE Trans. Comput., Vol. 20, pp. 290-298, March 1971.

4. _____, "Monotone Functions in Sequential Circuits", IEEE Trans. Comput., Vol. C-22, pp. 928-933, October 1973.

5. D. K. Pradhan and S. M. Reddy, "Techniques to Construct (2,1) Separating Systems from Linear Error Correcting Codes", IEEE Trans. Comput., Vol. C-25, pp. 945-949, September 1976.

6. J. H. Tracey, "Internal State Assignments for Asynchronous sequential Machines", IEEE Trans. Electron. Comput., Vol. EC-15, pp. 551-560. August 1966.

7. S. H. Unger, Asynchronous Sequential Switching Circuits, New York: Wiley-interscience 1969.

8. R. F. Spencer, Jr., "MOS Complexity Gates in Digital Systems Design", IEEE Comput. Group News, Vol. 2, pp. 47-56, September 1969.

9. R. Mine and Y. Koga, "Basic Properties and a Construction Method for Fail-Safe Logic Systems", IEEE Trans. Electron Comput., Vol. EC-16, pp. 282-289, June 1967.

10. Y. Tohma, Y. Ghyama and R. Sakai, "Realization of Fail-Safe Sequential Machines by Using k-out-of-n Code", IEEE Trans. Comput., Vol. C-20, pp. 1270-1275, November 1971.

11. R. Betancourt, "Derivation of Minimum Test Sets for Unate Logical Circuits", IEEE Trans. Comput., Vol. C-20, pp. 1264-1269, November 1971.

12. D. K. Pradhan, "Asynchronous State Assignments with Unateness Properties and Fault Secure Design", IEEE Trans. Comput., Vol. C-27, pp. 396-404, May 1978.

13. S. Das and H. Y. H. Chuang, "A Unified Approach to the Realization of Fail-Safe Sequential Machines", Digest of Fourth International Symposium on Fault Tolerant Computing, pp. 3-2 - 3-6, June 1974.

14. W. W. Peterson, E. J. Weldon, Jr., Error Correcting Codes, M.I.T. Press, Cambridge, Massachusetts, 1970.

15. T. R. N. Rao and S. D. Constantin, "Group Theoretical Codes for Binary Asymmetrical Channels," Technical Report CS 76014, Department of Computer Science, Southern Methodist University, Dallas, Texas, October 1976.

16. B. Bose and T. R. N. Rao, "On the Theory of Unidirectional Error Correction/Detection," Technical Report CS 7817, Department of Computer Science, Southern Methodist University, Dallas, Texas, September 1978.